UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA

PAULO MARTINS ENGEL

INBC: An Incremental Algorithm for Dataflow Segmentation Based on a Probabilistic Approach

Relatório de Pesquisa
RP 360

Órgão Financiador: CNPq

Porto Alegre, maio de 2009.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

INBC: um algoritmo incremental para segmentação de fluxos de dados baseado numa abordagem probabilística

RESUMO

Este relatório apresenta um novo algoritmo para a segmentação não-supervisionada de fluxos de dados baseado numa abordagem probabilística incremental. O algoritmo, chamado de INBC (Incremental Naïve Bayes Clustering), assume que as variáveis observadas são estatisticamente independentes, seguindo uma abordagem conhecida como Naïve Bayes. INBC cria e ajusta continuamente um modelo probabilístico consistente com todos os dados apresentados seqüencialmente. No domínio da robótica móvel, o algoritmo INBC detecta regularidades nos sinais sensoriais de entrada identificando segmentos de trajetórias correspondentes a conceitos de ordem mais alta como "parede à direita" ou "curva à esquerda".

O algoritmo adota um modelo de mistura de distribuições componentes que pode ser expandido para acomodar nova informação fornecida por um dado de entrada, ou reduzido, se forem identificadas componentes espúrias ao longo do processo de aprendizado. Por outro lado, cada dado assimilado pelo modelo contribui para a atualização dos seus parâmetros, baseada na maximização da verossimilhança de todos os dados apresentados. O ajuste dos parâmetros do modelo é baseado na acumulação de informações relevantes extraídas de cada dado observado. Os valores dos acumuladores são limitados a um máximo regulado por um fator de desconto, tornando possível o aprendizado incremental de seqüências de dados ilimitadas.

A aproximação adotada na atualização do modelo faz com que o INBC seja apropriado para a modelagem de ambientes não-estacionários, mas com parâmetros que variam lentamente com o tempo. Para mostrar a utilidade da abordagem adotada, são apresentados vários experimentos com dados simulados.

Palavras-Chave: aprendizado incremental, modelagem probabilística.

ABSTRACT

This technical report presents a new algorithm for unsupervised segmentation of data flows based on an incremental probabilistic approach. The algorithm called INBC (for Incremental Naïve Bayes Clustering) assumes that the observed variables are statistically independent, following an approach known as Naïve Bayes. INBC creates and continually adjusts a probabilistic model consistent to all sequentially presented data. In the domain of mobile robotics, INBC detects regularities in the sensor readings identifying trajectory segments corresponding to higher order concepts like "wall at right" or curve at left".

INBC adopts a mixture model of distribution components that can be expanded to accommodate new information from an input data point, or reduced if spurious components are identified along the learning process. On the other hand, each data point assimilated by the model contributes to the sequential update of the model parameters based on the maximization of the likelihood of the data. The parameters are updated through the accumulation of relevant information extracted from each data point. The accumulators are limited to a maximum value controlled by a discount factor, making possible the incremental learning of unbounded data streams.

The approximation used by INBC to update the model parameters makes it suitable to model non-stationary but slowly variable environments. INBC was tested with simulated data representing typical environments encountered in the considered application domains.

Keywords: incremental learning, Bayesian modeling.

1 INTRODUCTION

In autonomous robotics an important task is the detection of higher order concepts such as 'corners', 'walls' and 'corridors' from the sequence of noisy sensor readings of a mobile robot. The detection of these regularities in dataflow allows the robot to localize its position in the environment and to detect changes in the environment. In the past, different approaches were presented to this end, but they have scarce means to distinguish between relevant information and noise in the data stream, a problem frequently encountered in unsupervised learning known as the stability-plasticity dilemma. As a typical example of these approaches, Nolfi and Tani (1999) proposed a hierarchical architecture to extract regularities from time series, in which higher layers are trained to predict the internal state of lower layers when such states change significantly. In this approach, the segmentation was cast as a traditional error minimization problem, which favors the most frequent inputs, filtering out less frequent input patterns as being 'noise'. The result is that this system recognizes slightly differing walls, that represent frequent input patterns, as distinguish concepts, but is unable to detect corridors or corners that are occasionally (infrequently) encountered. On the other hand, focusing in change detection, Linåker and Niklasson (2000) proposed an adaptive resource allocating vector quantization (ARAVQ) network, which stores moving averages of segments of the data sequence as vectors allocated to output nodes of the network. New model vectors are incorporated to the model if a mismatch between the moving average of the input signal and the existing model vectors is greater than a specified threshold and a minimum stability criterion for the input signal is fulfilled. However, as other distance based clustering algorithm, this approach is not sensible to change in size and density of the clusters, producing models comparable to those computed by algorithms like the *k-means* (MACQUEEN, 1967).

In this work, we present a new algorithm based on a probabilistic framework to solve the problem of dataflow segmentation. The algorithm, called INBC (for Incremental Naïve Bayes Clustering), assumes that the observed variables are statistically independent, following an approach known as Naïve Bayes. The focus of this work is the so called unsupervised incremental learning, which considers building a model describing a data flow, where each data point is just instantaneously available to the learning system. In this case, the learning system needs to take into account the instantaneous data to update its model of the environment.

Our approach to incremental learning can be seen as an approximation of the traditional probabilistic methods that work on a fixed data set available at the beginning of the learning process. The problem must then be formulated from a Bayesian point of view and we discuss the assumptions made and the restrictions involved in this approximation.

Cluster analysis has a long tradition in machine learning literature, in the field of unsupervised learning. The well known *k-means* algorithm, for instance, represents a cluster as a mean of a subset of data (MACQUEEN, 1967). In this case, each data point must deterministically belong to one cluster. The membership of a data point to a cluster is decided by the minimum distance to the mean of the clusters. To compute the means, we average all data points belonging to every cluster, the number of clusters being fixed along all the learning process. For learning probabilistic models, the reference is the EM algorithm that follows a mixture distribution approach for probabilistic modeling explained in next section (DEMPSTER et al., 1977). This algorithm proceeds in two steps: an *estimation* step (E) that computes the probabilistic membership (the *posterior probability*) of every data to each component of the mixture model based on a current hypothesis (a set of parameters), followed by a *maximization* step (M) that updates the parameters of the current hypothesis based on the maximization of the *likelihood* of the data (the so called *maximum likelihood hypothesis*). Here the number of components is fixed and must be known at the start of the learning process. The parameters of each distribution are computed through the usual statistical point estimators, meaning that the complete training set must be previously known and fixed.

As the EM algorithm, our approach also follows the mixture distribution modeling. However, our model can be effectively *expanded* with new clusters as new relevant information is identified in the data flow. In our attempt to solve the problem of unsupervised incremental learning, we face then two problems: how to tackle the stability-plasticity dilemma and how to update the values of distribution parameters as new data points are sequentially acquired. Next section formalizes the probabilistic approach followed by INBC.

1.1 A Naïve Bayes approach for the mixture distribution model

INBC assumes that the probability density of the input data $p(\mathbf{x})$ can be modeled by a linear combination of k component densities $p(\mathbf{x}|\, j)$ corresponding to independent probabilistic processes, in the form

$$p(\mathbf{x}) = \sum_{j=1}^{M} p(\mathbf{x} \mid j) p(j) \qquad (1)$$

This representation is called a *mixture model* and the coefficients $p(j)$ are called the mixing parameters, related to the *prior* probability of $\mathbf{x}$ having been generated from component j of the mixture. The priors are adjusted to satisfy the constraints

$$\sum_{j=1}^{M} p(j) = 1 \qquad (2a)$$

$$0 \le p(j) \le 1 \qquad (2b)$$

Similarly, the component density functions $p(\mathbf{x}|\, j)$ are normalized so that

$$\int p(\mathbf{x} \mid j) d\mathbf{x} = 1 \qquad (3)$$

An important point in the INBC formulation is the assumption that the input variables, the components of $\mathbf{x}$, are mutually conditional independent. This is the same hypothesis followed by the so called *Naïve Bayes* classifier, meaning that the probability of observing the conjunction of the attributes is just the product of the probabilities for the individual attributes. Under this assumption, the probability of observing vector $\mathbf{x} = (x_1, ..., x_i, ..., x_d)$ belonging to the jth mixture component, is computed as the product

$$p(\mathbf{x} \mid j) = p(x_1 \mid j) \ldots p(x_i \mid j) \ldots p(x_d \mid j) \tag{4}$$

Moreover, for continuous attributes each individual component density $p(x_i \mid j)$ of distribution j is modeled by a one-dimensional normal Gaussian function that can be written in the form

$$p(x_i \mid j) = \frac{1}{\sqrt{2\pi\sigma_{ji}^2}} \exp\left\{ -\frac{(x_i - \mu_{ji})^2}{2\sigma_{ji}^2} \right\} \tag{5}$$

where μ_{ji} and σ_{ji}^2 are the *mean* and the *variance* respectively of the ith individual density of the jth mixture component.

One of the nice features of considering the components of **x** as mutually conditional independent is the fact that in (4) the individual components $p(x_i \mid j)$ can correspond either to continuous or to discrete attributes. Assume now that the discrete attribute x_i of vector **x** has one of V possible values from the set $\{x_{i1}, \ldots, x_{iv}, \ldots, x_{iV}\}$. For a specific data vector **x** with $x_i = x_{iv}$, $p(x_i \mid j) = p(x_{iv} \mid j)$, the probability of the corresponding attribute value. The $p(x_{iv} \mid j)$ can be estimated as the relative frequency of x_{iv} over the data, as discussed later in Chapter 2.

Although conditional independence is rarely true in real-world applications, Naïve Bayes models have been successfully used for clustering and classification (LOWD, D. & DOMINGOS, P., 2005). One reason for this is the fact that no matter how strong the dependences among attributes are the Naïve Bayes model can still be optimal if the dependences distribute evenly in classes, or if they cancel each other, what apparently frequently occurs (ZHANG, 2004).

2 AN INCREMENTAL MIXTURE MODEL

INBC adopts an *incremental* mixture distribution model, having special means to control the number of mixture components that effectively represent the so far presented data. The mixture model starts with one component with unity prior, centered at the first input data, with a baseline variance σ_{bi}^2 specified by default, i. e., $\mu_{1i} = x_i^1$, meaning the value of x_i for $t = 1$, and $\sigma_{1i}^2 = \sigma_{bi}^2$ for all individual components of vector $\mathbf{x}^1$.

The baseline variance can be specified by each individual attribute as a user defined fraction δ_i of the estimated overall variance of the corresponding attribute, computed from the range of these values according to

$$\sigma_{bi}^2 = \delta_i \, [\max(x_i) - \min(x_i)]^2 \qquad (6)$$

New components are added by demand. INBC uses a *minimum likelihood* criterion to recognize a vector $\mathbf{x}$ as belonging to a mixture component. For each incoming data point the algorithm verifies whether it *minimally fits* any mixture component. A data point $\mathbf{x}$ is not recognized as belonging to a mixture component j if its probability $p(\mathbf{x} \mid j)$ is lower than a previously specified *minimum likelihood-* (or *novelty-*) *threshold.* In this case, $p(\mathbf{x} \mid j)$ is interpreted as a *likelihood function* of the jth mixture component. If $\mathbf{x}$ is rejected by all density components, meaning that it bears new information, a new component is added to the model, appropriately adjusting its parameters. The novelty-threshold value affects the sensibility of the learning process to new concepts, with higher threshold values generating more concepts. By the NB assumption, however, $p(\mathbf{x} \mid j)$ is computed as the product of its attribute components. Therefore, it is more useful to specify a novelty threshold for the one-dimensional components. In this case, the user specifies a minimum value for the acceptable likelihood, τ_{nov}, as a *fraction* of the maximum value of the likelihood function. This is more intuitive for the user and makes the novelty criterion independent of the variance of the each attribute. Hence, a new mixture component is created when the values for all attributes of the instantaneous data point $\mathbf{x} = (x_1, ..., x_i, ..., x_d)$ match the *novelty criterion* written as

$$p(x_i \mid j) < \frac{\tau_{nov}}{\sqrt{2\pi\sigma_{ji}^2}} \qquad \forall i, \forall j \qquad (7)$$

Before explaining what happens when a new component is created, we present how INBC adjusts the values of the distribution parameters as new data points are sequentially acquired.

$$\mu_{ji}^{new} = \frac{\sum_{n=1}^{N} p^{old}(j \mid \mathbf{x}^n) x_i^n}{\sum_{n=1}^{N} p^{old}(j \mid \mathbf{x}^n)} \tag{11}$$

$$\sigma_{ji}^{2\,new} = \frac{\sum_{n=1}^{N} p^{old}(j \mid \mathbf{x}^n)\left(x_i^n - \mu_{ji}^{new}\right)^2}{\sum_{n=1}^{N} p^{old}(j \mid \mathbf{x}^n)} \tag{12}$$

$$p(j)^{new} = \frac{1}{N}\sum_{n=1}^{N} p^{old}(j \mid \mathbf{x}^n) \tag{13}$$

These new parameters are then used to update the *estimation* of the posteriors (step E), and the process is repeated until the new parameters eventually converge to the desired maximum likelihood solution, i.e. $\boldsymbol{\theta}^{new} = \boldsymbol{\theta}^*$. This procedure corresponds to the EM algorithm, which is guaranteed to decrease the value of $E(\boldsymbol{\theta})$ at each iteration, until a local minimum is found (DEMPSTER et al., 1977).

For discrete attributes, the maximization step corresponds to estimate the probability of observing each attribute *value* from a component j of the mixture. For a specific discrete attribute value x_{iv}, the probability $p(x_{iv} \mid j)$ can be estimated as the relative frequency of x_{iv} over the data vectors, weighted by the posterior probabilities that the corresponding data points were generated from that component j. This can be written as

$$p^{new}(x_{iv} \mid j) = \frac{\sum_{n \mid x_i = x_{iv}} p^{old}(j \mid \mathbf{x}^n)}{\sum_{n} p^{old}(j \mid \mathbf{x}^n)} \tag{14}$$

The sum at the numerator in (14) accumulates the posterior probabilities of the data vectors $\mathbf{x}^n$ that have value x_{iv} for the discrete attribute x_i.

INBC adopts an online approximation to (11), (12), (13) and (14) assuming that the model parameters change slowly over the time. The summations appearing in these expressions are computed recursively using the estimated posteriors and the attribute values of the current data point, as explained below.

At each time step t, when a data vector $\mathbf{x}^t$ is presented, INBC updates at first the variable sp_j, corresponding to the sum of posterior probabilities that *all* data presented so far were generated from component j, the so called 0*th-order moment of* $p(j \mid \mathbf{x})$ *over the data*, or simply the 0*th-order data moment for* j. The instantaneously presented data point $\mathbf{x}^t$ contributes to this sum with its posterior probability to the component j, written as

2.1 Model update by sequential assimilation of data points

An instantaneous data point that doesn't match the novelty criterion must be assimilated by the current mixture distribution causing an update in the values of its parameters due to the information it bears. INBC follows an incremental version for the usual iterative process to estimate the parameters of a mixture model based on two steps: an estimation step (E) and a maximization step (M). The update process begins computing the posterior probabilities of component membership for the data point, $p(j \mid \mathbf{x})$, the *estimation* step. These can be obtained through Bayes' theorem, using the current component-conditional densities $p(\mathbf{x} \mid j)$ and priors $p(j)$ as follows:

$$p(j \mid \mathbf{x}) = \frac{p(\mathbf{x} \mid j)p(j)}{\sum_{j=1}^{M} p(\mathbf{x} \mid j)p(j)} \qquad \forall j \tag{8}$$

The posterior probabilities can then be used to compute new estimates for the values of the mean μ_{ji}^{new} and variance $\sigma^{2}{}_{ji}^{new}$ of each component density $p(x_i \mid j)$, and the priors $p^{new}(j)$ in the *maximization* step. The usual maximization step updates the parameters of the current model based on the maximization of the likelihood of the data. Next, we briefly explain the ideas behind this procedure, considering the particular case of a mixture model composed only by one-dimensional Gaussian distributions $p(x_i \mid j)$, as defined by equations (1), (4) and (5).

The parameters $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_M)^T$, corresponding to the means, μ_{ji}, variances, $\sigma^2{}_{ji}$, and priors $p(j)$ of a mixture model involving one-dimensional Gaussian distributions $p(x_i \mid j)$, as is the case by INBC, can be estimated from a data set of N vectors, $\mathbf{X} = \{\mathbf{x}^1, ..., \mathbf{x}^n, ..., \mathbf{x}^N\}$ assumed to be drawn independently from this *mixture distribution*. In this case, the *likelihood* of $\boldsymbol{\theta}$ for the given $\mathbf{X}$, $L(\boldsymbol{\theta})$, is the joint probability density of the whole data set $\mathbf{X}$, given by

$$L(\boldsymbol{\theta}) \equiv p(\mathbf{X} \mid \boldsymbol{\theta}) = \prod_{n=1}^{N} p(\mathbf{x}^n \mid \boldsymbol{\theta}) = \prod_{n=1}^{N} \left[\sum_{j=1}^{M} p(\mathbf{x}^n \mid j)p(j) \right] \tag{9}$$

The technique of maximum likelihood sets the value of $\boldsymbol{\theta}$ by maximizing $L(\boldsymbol{\theta})$. This corresponds to choose the $\boldsymbol{\theta} = \boldsymbol{\theta}^*$, that is most likely to give rise to the observed data. Alternatively, the negative log-likelihood for the data set can be regarded as an error function, given by

$$E(\boldsymbol{\theta}) = -\ln L(\boldsymbol{\theta}) = -\sum_{n=1}^{N} \ln p(\mathbf{x}^n) = -\sum_{n=1}^{N} \ln \left[\sum_{j=1}^{M} p(\mathbf{x}^n \mid j)p(j) \right] \tag{10}$$

The expressions for the parameters $\boldsymbol{\theta}^*$ at the maximum of $L(\boldsymbol{\theta})$, equivalent to the minimum of (10), can be found iteratively by the EM-algorithm, given an initial guess for $\boldsymbol{\theta}$. At each iteration *new* parameters, $\boldsymbol{\theta}^{new}$, are computed through the following EM-update equations corresponding to the maximization step:

$$\sum_{n=1}^{t} p(j \mid \mathbf{x}^n) x_i^n = p(j \mid \mathbf{x}^t) x_i^t + \sum_{n=1}^{t-1} p(j \mid \mathbf{x}^n) x_i^n \tag{20}$$

Which is equivalent to the update of spx_{ji}, written as

$$spx_{ji}^{new} = prob(j \mid \mathbf{x}^t) x_i^t + spx_{ji}^{old} \tag{21}$$

Similarly, using spx_{ji}^{new} and sp_j^{new} in (11) a new estimate for the mean of the Gaussian representing the ith attribute of the jth component can be computed by

$$\mu_{ji}^{new} = \frac{p(j \mid \mathbf{x}^t) x_i^t + \sum_{n=1}^{t-1} p(j \mid \mathbf{x}^n) x_i^n}{p(j \mid \mathbf{x}^t) + \sum_{n=1}^{t-1} p(j \mid \mathbf{x}^n)} = \frac{spx_{ji}^{new}}{sp_j^{new}} \tag{22}$$

To update the value of the variance for the ith attribute of the jth component, we need to compute the sum of products of the posteriors by the square of the difference between the value of this attribute and its mean, the so called *2nd-order data moment about the mean for j*. Here, the new estimate for the mean μ_{ji}^{new} can be used as shown by

$$\sum_{n=1}^{t} p(j \mid \mathbf{x}^n)(x_i^n - \mu_{ji})^2 = p(j \mid \mathbf{x}^t)(x_i^t - \mu_{ji}^{new})^2 + \sum_{n=1}^{t-1} p(j \mid \mathbf{x}^n)(x_i^n - \mu_{ji})^2 \tag{23}$$

The variable sps_{ji} stores this summation, resulting in the following update expression

$$sps_{ji}^{new} = p(j \mid \mathbf{x}^t)(x_i^t - \mu_{ji}^{new})^2 + sps_{ji}^{old} \tag{24}$$

Using sps_{ji}^{new} and sp_j^{new} in (12) a new estimate for the variance of the Gaussian representing the ith attribute of the jth component can be computed according to

$$\sigma_{ji}^{2\,new} = \frac{p(j \mid \mathbf{x}^t)(x_i^t - \mu_{ji}^{new})^2 + \sum_{n=1}^{t-1} p(j \mid \mathbf{x}^n)(x_i^n - \mu_{ji})^2}{p(j \mid \mathbf{x}^t) + \sum_{n=1}^{t-1} p(j \mid \mathbf{x}^n)} = \frac{sps_{ji}^{new}}{sp_j^{new}} \tag{25}$$

$$\sum_{n=1}^{t} p(j \mid \mathbf{x}^n) = p(j \mid \mathbf{x}^t) + \sum_{n=1}^{t-1} p(j \mid \mathbf{x}^n) \qquad (15)$$

The left term and the last right term correspond respectively to the *new* and *old* values of the variable sp_j,

$$sp_j^{new} = p(j \mid \mathbf{x}^t) + sp_j^{old} \qquad (16)$$

This variable stores the current sum of posteriors that are used at the denominators of (11), (12) and (14), and at (13). As a matter of fact, sp_j represents an approximate estimate of the desired summation, as by each new presented data point the distribution parameters change. This actualization error is analog to the one made by the EM algorithm when the new parameter values are computed using posteriors computed by old parameters. In the case of the EM algorithm, this error is minimized through successive presentations of the whole data set, while in the case of the INBC the error is minimized through the successive presentation of new data. If the learning task is *episodic*, meaning that it evolves in cycles from a starting state to a terminal state, it is very likely that all distribution components of the INB model will be properly updated, equivalently to the repeated presentations of the whole data set for the EM algorithm. However, as the posteriors of the past observations are not updated by INBC, in *continuing*, *non-episodic* tasks this approximation may fail if the model parameters change rapidly. This makes INBC suitable either for episodic tasks as well for infinite horizon tasks in non-stationary but slowly variable environments.

Using sp_j^{new} in (13), a new estimate for the prior probability for the component j can be computed by

$$p(j)^{new} = \frac{1}{t}\sum_{n=1}^{t} p(j \mid \mathbf{x}^n) = \frac{1}{t} sp_j^{new} \qquad (17)$$

Note that (17) implies that the current number of presented data points, equivalent to t, must be stored. However, the total number of presented data points can be computed summing all sp_j, and so no extra counter is needed to this end, as can be shown by

$$t = \sum_{j=1}^{M} sp_j \qquad (18)$$

So, equation (17) can be replaced by

$$p(j)^{new} = \frac{sp_j^{new}}{\sum_{j=1}^{M} sp_j^{new}} \qquad (19)$$

Similarly, the variable spx_{ji} stores the current sum of products of posterior probabilities for component j by the respective values of attribute i for all data presented so far, the so called *1st-order data moment for j*. The instantaneously presented data point $\mathbf{x}^t$ contributes to this sum with the product of its posterior probability to the component j by the value of its attribute i according to

Consider now the case that **x** has a discrete attribute x_i. At an instant t, a data vector $\mathbf{x}^t$ with $x_i = x_{iv}$ contributes with its posterior probability $p(j \mid \mathbf{x}^t)$ to the counter corresponding to the value of its attribute at component j. So, to compute $p(x_{iv} \mid j)$ we need a variable sp_{jiv} for each discrete value of this attribute, summing up the posterior probabilities for data points containing the discrete attribute value x_{iv} as shown by

$$sp_{jiv}^{new} = p(j \mid \mathbf{x}^t) + sp_{jiv}^{old} \qquad \forall \mathbf{x}^t \mid x_i^t = x_{iv} \tag{26}$$

The probability for the attribute value x_{iv} is then computed as the relative frequency of the accumulated values according to

$$p(x_{iv} \mid j)^{new} = \frac{\sum_{n \mid x_i = x_{iv}} p(j \mid \mathbf{x}^n)}{\sum_{n} p(j \mid \mathbf{x}^n)} = \frac{sp_{jiv}^{new}}{sp_j^{new}} \tag{27}$$

Alternatively, (27) can be replaced by the so called *Laplace estimator* that avoids empty accumulators and the undesirable zero values for these probabilities, leading to a forced zero $p(\mathbf{x} \mid j)$ through (4). In this case, every accumulator for each discrete attribute value is incremented by one, and the corresponding number of attribute values, n_{iv}, is added to the denominator as shown by

$$p(x_{iv} \mid j)^{new} = \frac{sp_{jiv}^{new} + 1}{sp_j^{new} + n_{iv}} \tag{28}$$

These equations formalize the maximization step of INBC.

2.1.1 Creating a new component: novelty and stability criteria

To create a new component, INBC uses the novelty criterion given by (7), but also a stability criterion that tests if there is already a recent created component that should assimilate the current presented data point. The stability criterion is based on the assumption explained before that the model parameters should not change rapidly. This also means that there is a *stable* dominant component for the current data, which is equivalent to say that the environment can be decomposed in stable contexts, the *concepts* of the environment. To this end, we store the *age* of each model component j, age_j, consisting of the number of data that have been presented to the learning system since the component was created. A new component is created only if there is no model component whose age is less than a specified threshold τ_{age}, i. e., the stability criterion can be defined by

$$age_j > \tau_{age} \qquad \forall j \tag{29}$$

Then, if the data change too fast, these data points are assimilated by the current model as noise and no new component is created. As a matter of fact, the stability criterion avoids the successive creation of components in a noisy environment, but it cannot avoid the creation of an eventual spurious component. However, a spurious component can be readily identified if its sp_j remains very small, bellow a specified parameter *spmin*, after some time steps since its creation, given by the parameter *agemin*. Once identified, a spurious component can be deleted from the current model. The condition to identify a spurious component can be written as

$$\text{IF } age_j > agemin \cap sp_j < spmin \text{ THEN } spurious_j = 1$$

where $spurious_j$ is a flag identifying component j as spurious.

In the experiments presented in Chapter 3, $\tau_{age} = 2$, $spmin = 3$ and $agemin = 5$.

Whenever a data point $\mathbf{x}^t$ matches the novelty criterion given by (7), and the stability criterion is fulfilled, a new component $M = M^{old} + 1$ is created, centered at that data vector and with the baseline variance, i. e., $\mu_{M,i} = x_i^t$ and $\sigma_{M,i}^2 = \sigma_{bi}^2$. Since sp_M is initialized with 1, the prior for this new component is adjusted according to (19) to:

$$p(M) = \frac{1}{\sum_{j=1}^{M} sp_j^t} \tag{30}$$

However, we must adjust all $p(j)$ to satisfy the constraint (2a). Here we could apply different heuristics, but we found that a simple overall normalization produced good results in our experiments, meaning that the priors are adjusted by

$$p(j)^{new} = \frac{p(j)^{old}}{\sum_{j=1}^{M} p(j)^{old}} \tag{31}$$

Here, $p(j)^{old}$ represents the prior for the jth component, including the new one, immediately after creating a new component, and $p(j)^{new}$ represents this prior after normalization.

2.2 Incremental learning from unbounded data streams

INBC was designed to process data streams creating on-line models of eventually huge amounts of data based on accumulation of information collected by individual observations. This unbounded accumulation may lead to an eventual saturation of the variables that store the corresponding summations, depending on the specific limitation of the available numeric representation. To avoid this problem, a discount factor given by a parameter α is applied to the instantaneous value of the corresponding variable. For the accumulator of the posteriors, for example, we obtain:

$$sp_j^{new} = p(j \mid \mathbf{x}^t) + \alpha sp_j^{old} \tag{32}$$

Since the instantaneous sum of $p(j \mid \mathbf{x}^t)$ over all components j is one, if we choose $0 < \alpha < 1$ as

$$\alpha = 1 - \frac{1}{n_{\max}} \tag{33}$$

where n_{max} is a large number within the available numeric representation, the accumulation over the time of all sp_j according to the recursive equation (32) is bounded to n_{max}, as shown by

$$\lim_{t \to \infty} \sum_{n=1}^{t} \sum_{j=1}^{M} sp_j^n = n_{\max} \tag{34}$$

Similarly, we apply the discount factor α to all other accumulators resulting in

$$spx_{ji}^{new} = prob(j \mid \mathbf{x}^t) x_i^t + \alpha spx_{ji}^{old} \tag{35}$$

$$sps_{ji}^{new} = p(j \mid \mathbf{x}^t)(x_i^t - \mu_{ji}^{new})^2 + \alpha sps_{ji}^{old} \tag{36}$$

$$sp_{jiv}^{new} = p(j \mid \mathbf{x}^t) + \alpha sp_{jiv}^{old} \qquad \forall \mathbf{x}^t \mid x_i^t = x_{iv} \tag{37}$$

The long term behavior of these recursive equations can be readily understood considering that the model is locally stable, meaning that the expressions (22), (25) and (27) for the corresponding parameters are valid. In this case, from (34) follows that these equations are bounded to the product of n_{max} with the corresponding parameter, as shown by

$$\lim_{t\to\infty}\sum_{n=1}^{t}\sum_{j=1}^{M} spx_{ji}^{n} = \lim_{t\to\infty}\sum_{n=1}^{t}\sum_{j=1}^{M} \mu_{ji} sp_{j}^{n} = \mu_{ji} n_{\max} \tag{38}$$

$$\lim_{t\to\infty}\sum_{n=1}^{t}\sum_{j=1}^{M} sps_{ji}^{n} = \lim_{t\to\infty}\sum_{n=1}^{t}\sum_{j=1}^{M} \sigma_{ji}^{2} sp_{j}^{n} = \sigma_{ji}^{2} n_{\max} \tag{39}$$

$$\lim_{t\to\infty}\sum_{n=1}^{t}\sum_{j=1}^{M} sp_{jiv}^{n} = \lim_{t\to\infty}\sum_{n=1}^{t}\sum_{j=1}^{M} p(x_{iv} \mid j) sp_{j}^{n} = p(x_{iv} \mid j) n_{\max} \tag{40}$$

A large n_{max} is desirable to keep the overall strategy stable. In our experiments, we observed that $n_{max} \geq 10^6$ produced negligible effects in the generated models.

Moreover, we observe from these long-term equations that when the model is at a local minimum, its parameters are dependent from the *relation* among the values of the accumulators but not on the values themselves. So, to better explore the available numeric range keeping the system responsive to new components, whenever long-term mode is reached, identified as a fraction β of n_{max}, the accumulators are restarted to a fraction γ of their corresponding value, i.e.

Whenever

$$\sum_{j=1}^{M} sp_{j} \geq \beta n_{\max} \tag{41}$$

then,

$$sp_{j}^{new} = \gamma sp_{j}^{old} \tag{42}$$

$$spx_{ji}^{new} = \gamma spx_{ji}^{old} \tag{43}$$

$$sps_{ji}^{new} = \gamma sps_{ji}^{old} \tag{44}$$

$$sp_{jiv}^{new} = \gamma sp_{jiv}^{old} \tag{45}$$

In the experiments presented in Chapter 3, we used $\beta = 0.8$ and $\gamma = 0.5$.

3 EXPERIMENTS

To highlight the main features of the algorithm, we present a series of experiments involving simulated data related to the domain of mobile robots trajectories. In the first two experiments the data consist of a sequence of vectors with 2 continuous components representing the instant position of a mobile robot moving along a corridor and a rectangular environment, respectively. Although in real mobile robotics we can rarely count on the actual instant position of the robot, these data were used because the resulting computed models are more easily interpreted than the more realistic multidimensional sensor data of a real robot, used in the next experiments. Then, in the next experiments, the data consist of 4 continuous values corresponding to the readings of the sonar sensors of a Pioneer mobile robot. In all experiments, the data was corrupted with noise.

3.1 Moving along a corridor: models with one component

In this experiment, INBC receives a sequence of data points corresponding to the noisy observations of the (x, y) position of a robot moving along a corridor following a loop with a halfway intersection, as shown in Figure 3.1. In this figure a square corresponds to a data point representing the real position of the robot in the range [−100, +100; −2, +2] added with random noise from a normal distribution with mean zero and standard deviation 0.1. Walls are represented by two parallel thick lines at $y = \pm 20$. A complete lap starts and ends at (−100, 0) generating 200 data points, but the robot can iterate many laps, generating data sequences of different lengths. In the experiment shown in Figure 3.1, the robot made just a single lap and INBC created just one cluster representing these data. The central dot in this figure represents the position of the cluster generated by INBC. The two parallel thin lines at ±5.3 represent the y-positions of the borders of the cluster according to the novelty criterion while the corresponding x-positions fall out of the figure and are not visible. Table 3.1 shows the clusters parameters after a single iteration through the data sequence and the corresponding statistics computed for these data. We can see that the values of the cluster parameters computed by INBC and those from the statistics for these data are very close. In this table the value for the other parameters of the algorithm used in this experiment are also included.

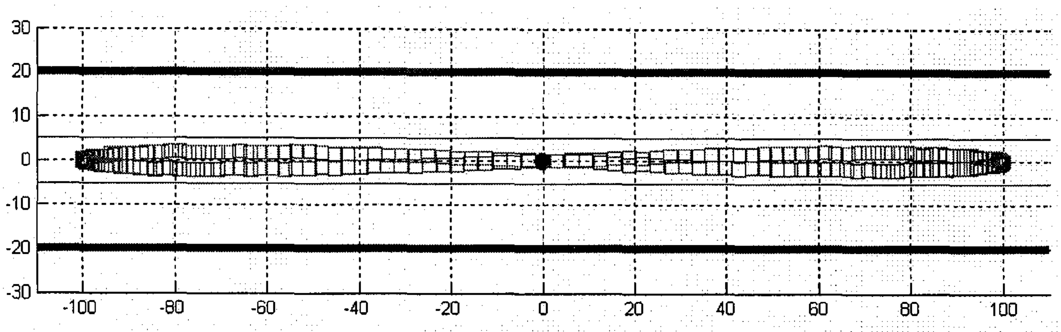


Figure 3.1: The dot at the center of the figure represents the position of the mean of the single cluster generated by INBC after a complete lap of the robot around the corridor. The squares represent the data points presented to the algorithm.

Table 3.1: Distribution parameters computed by INBC and statistics

τ_{nov}: 0.001 δ: 0.1 $n_{\max}$: 10^{+6}				
	Cluster 1 (INB)		Statistics	
	Mean	Std Dev	Mean	Std Dev
x	−0.0069	70.4716	−0.0069	70.8793
y	0.0083	1.4230	0.0083	1.4319

3.1.1 Handling large data sequences

In this experiment, we verify the effect of restarting the counters to a fraction γ of the current value when it reaches $\beta n_{\max}$, comparing the results obtained without restarting, and with a discount factor of unity. In this case, the robot iterates 5000 laps through the environment, generating 10^6 data points that is the same value of *nmax*. Table 3.2 shows the results obtained restarting the accumulators with $\beta = 0.8$ and $\gamma = 0.5$. These results show that the models parameters computed restarting the accumulators to 50% of their original value when the sum of *sp* reaches 80% of $n_{\max}$ are essentially the same as those computed without decay and restarting, indicating that the strategy of decaying and restarting the accumulators is effective for learning from unbounded data streams.

Table 3.2: Distribution parameters and final values of the accumulators computed by INBC with and without restart

τ_{nov}: 0.001 δ: 0.1 $n_{\max}$: 10^{+6} $\beta = 0.8$ $\gamma = 0.5$					
with restart	Mean	Std Dev	*sp*	*spx*	*sps*
x	−0.0001	70.7105	632120.74	−52.42	3160586092.22
y	−0.0001	1.4177		−52.57	1270555.81
without decay/restart	Mean	Std Dev	*sp*	*spx*	*sps*
x	−0.0003	70.7104	1000000.00	−262.23	4999959219.43
y	−0.0001	1.4176		−64.42	2009647.56

3.2 Moving changing direction: models with many components

In this experiment, the robot moves along a rectangular environment, experiencing abrupt changes in direction. Figures 3.2 shows the results obtained after a complete lap through the environment. Table 3.3 shows the parameters of the model computed by INBC for the data presented in these figures. INBC created 4 clusters, one for each trajectory segment. As a result of the Naïve Bayes approach followed by INBC, each distribution component is aligned with the coordinate axes, as can be seen in this figure. The colored dots in the center of each segment represent the position of the mean of the corresponding cluster. The rectangles surrounding each center represent the borders of the corresponding cluster according to the novelty criterion.

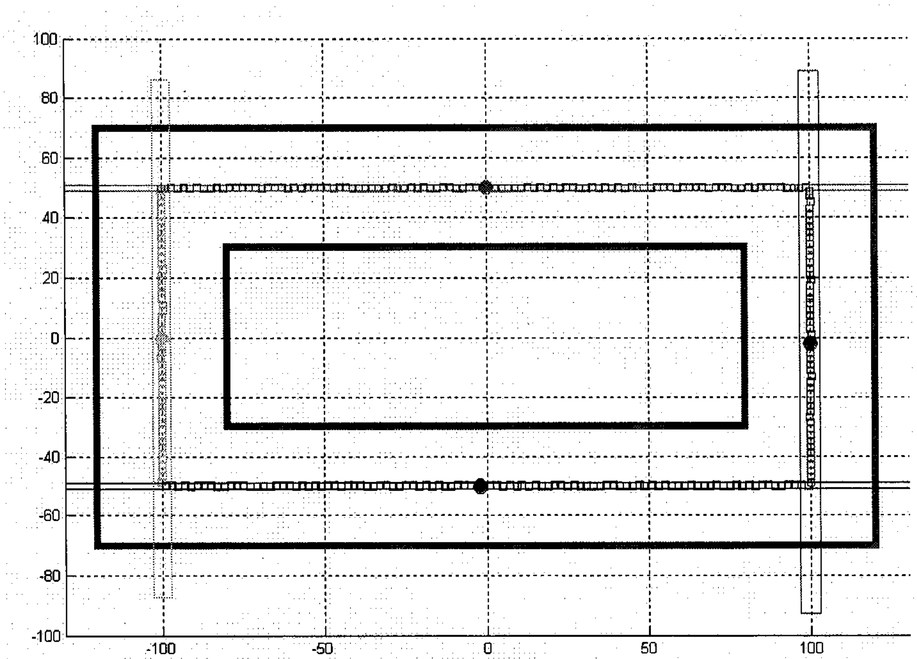


Figure 3.2: Results obtained with INBC after a single lap through the environment.

Table 3.3: Distribution parameters computed by INBC for the experiment shown in Figure 3.2

τ_{nov}: 0.01 δ: 0.02 $n_{\max}$: 10^{+6} $\beta = 0.8$ $\gamma = 0.5$							
C1 p: 0.3336		Mean	StdDev	C3 p: 0.3269		Mean	StdDev
	x_1	0.0140	57.8698		x_1	−2.0729	56.6961
	x_2	50.0002	0.3457		x_2	−50.0030	0.3362
C2 p: 0.1740		Mean	StdDev	C4 p: 0.1655		Mean	StdDev
	x_1	99.9093	1.0236		x_1	−99.9824	0.9021
	x_2	−1.9513	29.9087		x_2	−0.4389	28.5697

Figure 3.3 shows the results obtained applying the EM algorithm to these data setting the number of clusters to 4. Table 3.4 shows the parameters of the model computed by the EM algorithm. From these figures and tables, we can see that the model computed by INBC is a very good approximation to the model computed by the EM algorithm.

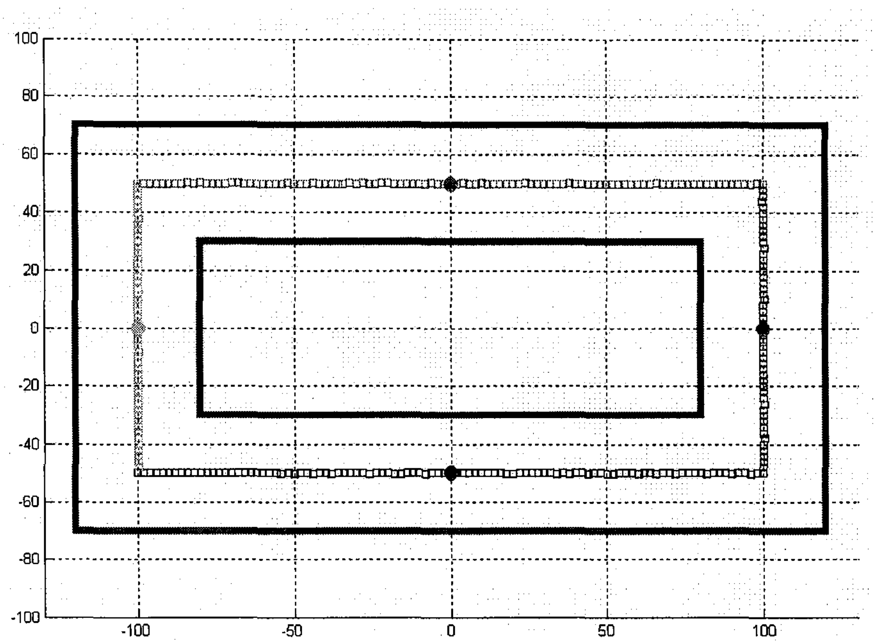


Figure 3.3: Results obtained with the EM algorithm.

Table 3.4: Distribution parameters computed by the EM algorithm for the experiment shown in Figure 3.3

EM algorithm with 4 clusters							
C1 p: 0.3336		Mean	StdDev	C3 p: 0.3336		Mean	StdDev
	x_1	0.1130	57.7856		x_1	0.0206	57.7850
	x_2	50.0016	0.0963		x_2	−50.0044	0.0982
C2 p: 0.1662		Mean	StdDev	C4 p: 0.1666		Mean	StdDev
	x_1	100.0018	0.1098		x_1	−99.9808	0.0974
	x_2	−0.0376	28.8064		x_2	0.0666	28.8586

3.3 Detecting concepts from sonar signals

In these experiments, the data consist of a sequence of 4 continuous values (s_1, s_2, s_3, s_4) corresponding to the readings of the sonars located at the left/right side (s_1, s_4) and at $-10°/+10°$ from the front (s_2, s_3) of a robot, generated using the Pioneer simulator software ARCOS (Advanced Robot Control & Operations Software). Figure 3.4 shows the segmentation of the trajectory obtained by INBC when the robot follows the walls of a square room. INBC created two clusters corresponding to the concepts of "wall at left" and "curve at right". The colored filled dots in this figure correspond to the location where each cluster was created. A square represents a robot position and has the same color of the cluster with the largest posterior probability for the corresponding data point. Table 3.5 shows the parameters of the model computed by INBC. For these data, 5000 corresponds to the maximum value of the sensor, indicating that no obstacle was detected in the corresponding direction. Figure 3.5 (a) shows the plots of the means appearing in Table 3.5 for the pair of signals of the sonars at the left and right side (s_1, s_4) and, (b), in the front (s_2, s_3). In this figure a rectangle represents the borders of the corresponding cluster according to the novelty criterion.

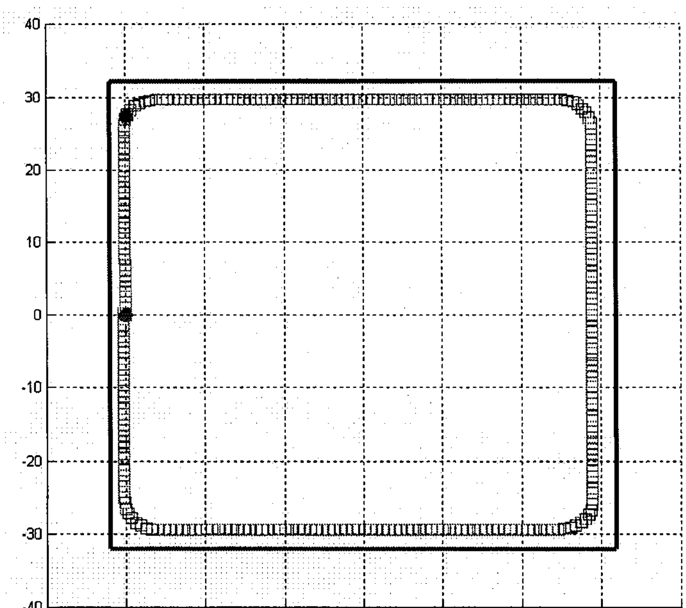
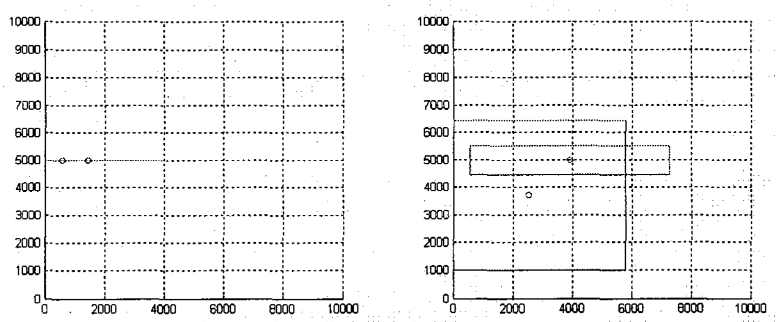


Figure 3.4: Each colored dot corresponds to the position where a cluster was created by INBC. A square represents a robot position and has the same color of the cluster with the largest posterior probability for the corresponding data. Red segments correspond to the concept "wall at left", blue segments to the concept "curve at right".

From the means presented in Table 3.5 we can see that the concept "wall at left" is mainly related to a small value of s_1, meaning a close obstacle detected at left, whereas "curve at right" is related to the detection of (not so close) obstacles at left and at $-10°$ in the front.

Table 3.5: Distribution parameters computed by INBC for the experiment shown in Figure 3.4

τ_{nov} : 10^{-6} δ: 0.05 $n_{\max}$: 10^{+6} β = 0.8 γ= 0.5			
Cluster 1 - red ("wall at left") Prior 0.9257			
Mean s_1 (left)	592	Std. Dev s_1	109
Mean s_2 (front –10°)	3927	Std. Dev s_2	639
Mean s_3 (front +10°)	4987	Std. Dev s_3	101
Mean s_4 (right)	5000	Std. Dev s_4	0.01
Cluster 2 - blue ("curve at right") Prior 0.0743			
Mean s_1 (left)	1442	Std. Dev s_1	506
Mean s_2 (front –10°)	2553	Std. Dev s_2	619
Mean s_3 (front +10°)	3710	Std. Dev s_3	514
Mean s_4 (right)	5000	Std. Dev s_4	0.01



(a) (b)

Figure 3.5: The components of the mean of each cluster created by INBC for the data shown in Figure 3.4. The data components are grouped in pairs, (s_1, s_4) corresponding to left/right sonars in (a), (s_2, s_3) corresponding to the –10°/+10° front sonars in (b). The rectangles show the borders of each cluster according to the novelty criterion.

Figure 3.6 shows the segmentation of the trajectory of a robot following the walls in an environment with two different sized rooms connected by a short corridor. In this case, INBC created five clusters corresponding to the concepts "wall at right", "corridor", "obstacle front/curve at left", "corridor at right" and "curve at right". Table 3.6 shows the parameters of the model computed by INBC and Figure 3.7 shows the plots of the means of each cluster for these data. Here too, we interpreted each concept extracted by INBC from the values of the means presented in Table 3.6. In this case, "wall at right" corresponds to obstacle at right; "corridor" means obstacles at right and at left. The concept "obstacle front/curve at left" corresponds to the detection of a close obstacle at right and an obstacle not so close in the front. From the point of view of the means, "corridor at right" corresponds to the detection of a relative close obstacle in the front and no obstacles at both sides. Finally, "curve at right" stores the sensor readings at the positions where the robot makes a turn to the right by entering or exiting the short corridor. Note that here we chose a more comprehensible concept based on the action that the robot takes, but that is not included by simplicity in this experiment. In more realistic experiments, we would include the action as an extra input variable to the learning system.

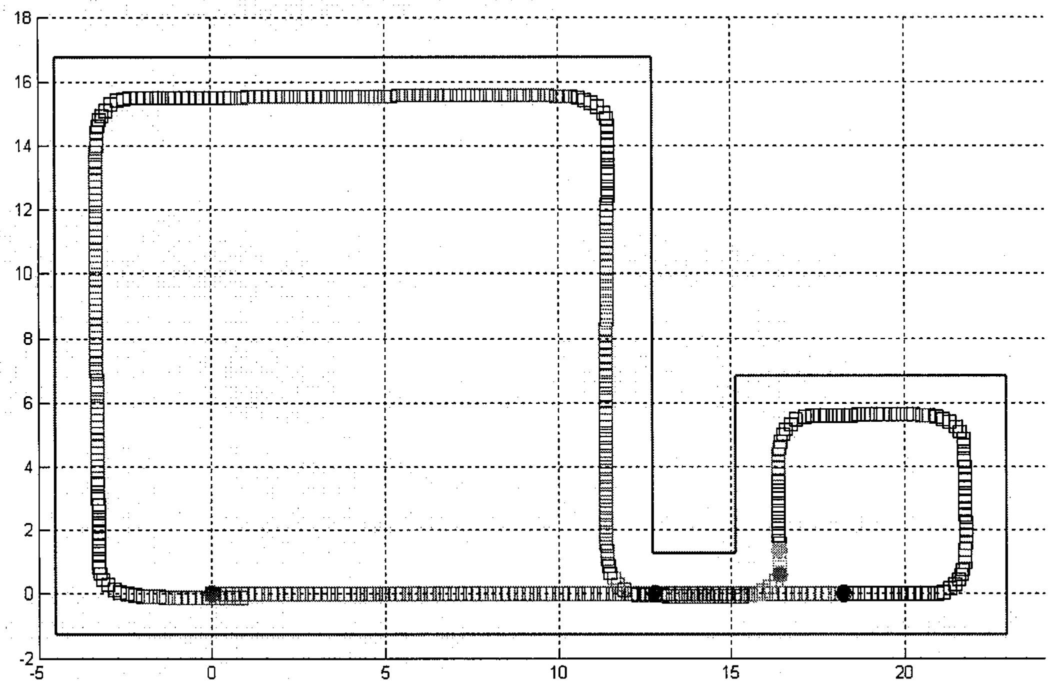


Figure 3.6: Each colored dot corresponds to the position where a cluster was created by INBC. A square represents a robot position and has the same color of the cluster with the largest posterior probability for the corresponding data.

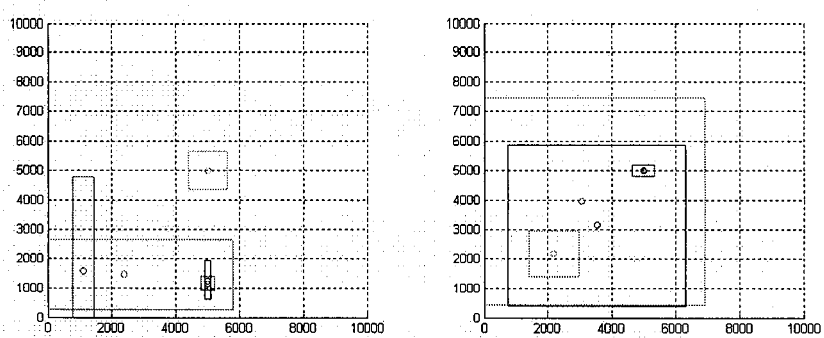


Figure 3.7: The components of the mean of each cluster created by INBC for the data shown in Figure 3.6. The 4 data components (s_1, s_2, s_3, s_4), are grouped in pairs, (s_1, s_3) in (a), (s_2, s_4), in (b).

Table 3.6: Distribution parameters computed by INBC for the experiment shown in Figure 3.6

τ_{nov} : 0.02 δ: 0.05 $n_{\max}$: 10^{+6} β = 0.8 γ = 0.5			
Cluster 1 - red ("wall at right") Prior 0.5797			
Mean s_1 (left)	4999	Std. Dev s_1	72
Mean s_2 (front −10°)	4999	Std. Dev s_2	24
Mean s_3 (front +10°)	4999	Std. Dev s_3	25
Mean s_4 (right)	1157	Std. Dev s_4	86
Cluster 2 - blue ("corridor") Prior 0.0664			
Mean s_1 (left)	1113	Std. Dev s_1	116
Mean s_2 (front −10°)	4986	Std. Dev s_2	130
Mean s_3 (front +10°)	5000	Std. Dev s_3	66
Mean s_4 (right)	1581	Std. Dev s_4	1143
Cluster 3 - black (" obstacle front/curve at left") Prior 0.3256			
Mean s_1 (left)	5000	Std. Dev s_1	36
Mean s_2 (front −10°)	3536	Std. Dev s_2	995
Mean s_3 (front +10°)	3148	Std. Dev s_3	980
Mean s_4 (right)	1259	Std. Dev s_4	234
Cluster 4 - cyan ("corridor at right") Prior 0.0078			
Mean s_1 (left)	5000	Std. Dev s_1	221
Mean s_2 (front −10°)	2188	Std. Dev s_2	282
Mean s_3 (front +10°)	2175	Std. Dev s_3	277
Mean s_4 (right)	5000	Std. Dev s_4	228
Cluster 5 - magenta ("curve at right") Prior 0.0205			
Mean s_1 (left)	2372	Std. Dev s_1	1221
Mean s_2 (front −10°)	3063	Std. Dev s_2	1380
Mean s_3 (front +10°)	3962	Std. Dev s_3	1254
Mean s_4 (right)	1465	Std. Dev s_4	422

From these results, we can see that INBC has segmented the trajectories according to a set of concepts that are useful to locate the robot in the environment. In the future, we will include the robot actions as input variables and investigate the use of the learned concepts for task planning.

4 CONCLUSION

In this technical report we presented a new algorithm for dataflow segmentation based on an incremental probabilistic model, called INBC, for Incremental Naïve Bayes Clustering. The adopted mixture distribution model is expanded based on a novelty criterion, and a stability criterion avoids the successive creation of spurious components in a noisy environment. The model is updated by sequential assimilation of data points. The update equations are incremental approximation versions of the update equations of the well known EM-algorithm based on accumulation of information collected by individual observations. We showed that the corresponding accumulators used to update the parameters of the model are bounded to a maximum value controlled by a discounted factor, allowing INBC to be used for modeling unbounded data streams. Experiments with simulated data streams of sonar readings of a mobile robot showed that INBC could efficiently segment trajectories detecting higher order concepts like "wall at right" and "curve at left". Future developments will expand the current approach to a hierarchy of segmentation layers.

REFERENCES

DEMPSTER, A. P., LAIRD, N. M., and RUBIN, D. B. Maximum likelihood from incomplete data via the EM algorithm. **Journal of the Royal Statistical Society**, Series B 39 (1), pp. 1-38, 1977.

LINÅKER, F. & NIKLASSON, L. Time series segmentation using an adaptive resource allocating vector quantization network based on change detection. In: INTERNATIONAL JOINT CONFERENCE ON NEURAL NETWORKS, IJCNN 2000. **Proceedings ...**, pp. 323-328. Los Alamitos. IEEE Computer Society, 2000.

LOWD, D. & DOMINGOS, D. In: INTERNATIONAL CONFERENCE ON MACHINE LEARNING, 22nd, 2005. **Proceedings...**, Bonn Germany, pp. 529-536.

LUTTRELL, S. P. The Partitioned Mixture Distribution: An Adaptive Bayesian Network for Low-Level Image Processing. **IEE Proceedings on Vision, Image and Signal Processing**, 141 (4), pp. 251-260, 1994.

MACQUEEN, J. Some Methods for classification and analysis of multivariate observations. In: BERKELEY SYMPOSIUM ON MATHEMATICAL STATISTICS AND PROBABILITY, 5th , 1967. **Proceedings...**, pp. 281-297. University of California Press, 1967.

NOLFI, S. & TANI, J. Extracting Regularities in Space and Time Through a Cascade of Prediction Networks: The Case of a Mobile Robot Navigating in a Structured Environment. **Connection Science**, v. 11, n.2, 1999.

ZHANG H. The Optimality of Naive Bayes. In: FLORIDA ARTIFICIAL INTELLIGENCE RESEARCH SOCIETY CONFERENCE, 7th, 2004. **Proceedings...**, pp. 562-567. The AAAI Press, 2004.